**TP DOCKERFILE**

Dans ce TP vous verrez la création d'une image avec des configurations personnalisées par l'écriture d'un Dockerfile. Vous devez disposer d'une vm Linux avec Docker installé et d'un container utilisant l'image Nginx.

**I/Mise en place**

Commencez par récupérer les fichiers /etc/nginx/conf.d/default.conf et /usr/share/nginx/html/index.html de votre container via la commande docker cp. Créez ensuite l'arborescence suivante sur votre vm :

/home/<votre\_utilisateur>/nginx/conf/  
/home/<votre\_utilisateur/nginx/code/intra  
/home/<votre\_utilisateur/nginx/code/wiki

Placez dans le dossier conf le fichier default.conf et placez le fichier index.html dans code/intra et également dans code/wiki. Modifiez les fichier index.html de chaque dossier pour qu'ils affichent "bienvenue sur l'intranet" ou "bienvenue sur le wiki" en fonction du répertoire dans lequel vous vous trouvez. Renommez le fichier default.conf en intra.conf et dupliquez le avec le nom wiki.conf. Modifiez maintenant les fichiers intra.conf et wiki.conf pour qu'ils indiquent que les répertoires racines de leurs codes soient situés dans /wiki et /intra et indiquez pour le site wiki qu'il écoute sur le port 8080. Avec cette configuration le service ira chercher dans les repertoires /wiki et /intra du container pour afficher les fichiers de codes html des sites.

Nous allons donc pouvoir passer à la rédaction du fichier Dockerfile qui va nous permettre de générer une image se basant sur l'image Nginx mais avec nos modifications. Ces modifications comportent :

* La création des dossiers /wiki et /intra
* La mise en place des fichiers de configurations précédemment modifiés
* L'ouverture des ports 80 et 8080 pour nos deux sites

Nous ne déplacerons pas les fichiers de codes par le dockerfile, nous utiliserons plutôt le mapping de volumes afin que les modifications des codes sources puissent être réalisés de manière dynamique.

**II/Rédaction du Dockerfile**

Maintenant que vos documents sont prêts nous allons passer à la rédaction du Dockerfile permettant de générer votre image. Une fois que votre image sera généré vous pourrez l'utiliser en lieu et place de l'image Nginx.

Un Dockerfile commence toujours par l'instruction FROM indiquant quelle image nous allons utiliser comme base pour la nôtre. Nous pourrions utiliser une image debian et y installer les différents paquets nécessaires à notre service. Etant donné que nous disposons déjà d'une image Nginx fonctionnelle c'est celle-ci que nous utiliserons pour créer la nôtre. Nous indiquerons ensuite, par l'instruction RUN, les commandes à effectuer pour créer nos répertoires. Nous finirons le document par l'instruction EXPOSE qui nous permet d'exposer les ports de l'image. /!\ Il ne s'agit pas du port de votre VM mais bien le port de l'image que vous exposez. Il faudra par la suite attribuer des ports de votre VM aux ports exposés de votre container.

Enregistrez votre fichier dans le répertoire nginx que vous avez créés et appelez le Dockerfile.

**III/Déploiement de l'image**

En étant dans le repertoire qui contient votre Dockerfile vous pouvez utiliser la commande   
docker build –t <nom\_a\_donner\_a\_votre\_image> . (Attention à ne pas oublier le . qui permet d'indiquer que le fichier se trouve là où vous êtes).

En sortie de cette commande vous devriez voir les différentes étapes de votre Dockerfile et enfin les syntax :  
Successfully built xxxxx  
Successfully tagged xxxxxx

Ceci indique que la création de l'image s'est correctement effectuée. Si vous obtenez une erreur modifiez votre Dockerfile en conséquence.

Vous pouvez utiliser la commande docker image ls afin de visualiser les images présentes sur votre machine. Vous y trouverez normalement les images hello-world, nginx, votre image et toute autre image que vous auriez utilisé pour instancier un container. Si vous avez mal configuré votre image vous avez toujours la possibilité de la supprimer avec la commande   
docker image rm <nom\_de\_l'image> attention toutefois si vous essayez de supprimer une image utilisé par un container vous aurez une erreur.

**IV/Utilisation de l'image**

Maintenant que votre image est créée vous pouvez vous en servir pour démarrer un nouveau container. Démarrez-en donc un détaché de votre terminal, en lui donnant un nom, en faisant bien attention à exposer les ports et en utilisant l'image que vous avez créé. Nous allons cette fois ajouter une option à cette commande de démarrage de container afin de mapper les répertoires contenant les codes sources de vos sites à leurs cibles dans le container. Cela va agir comme un lien symbolique entre le répertoire de votre machine physique et un répertoire du container. Ainsi lorsqu'on modifiera un fichier de ce répertoire sur la VM cela se répercute dans le container. C'est avec l'option   
-v <source>:<destination> que nous effectuerons cette action.

Si tout s'est bien passé dans la création de l'image et dans son utilisation pour instancier un container vous pouvez accéder au site intranet et au site wiki par les différents ports que vous avez attribués.

Pour finir créez l'arborescence /home/<votre\_utilisateur>/bind/data/. Dans le dossier bind créé, rédigez un fichier Dockerfile qui va se baser sur l'image sameersbn/bind, changer le mot de passe d'accès à la page d'administration du service et qui va exposer les mêmes ports de l'image de base.

Compilez votre image, déployez la dans un container et testez que vous pouvez bien accéder à la page d'administration en utilisant le mot de passe que vous avez définis.